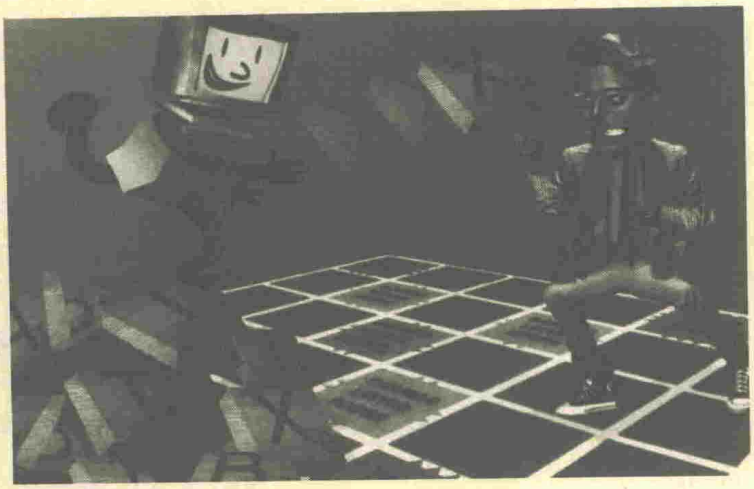
อาจหาญ สัตยารักษ์

SOFT MAIL

โอนภาพ PCX มาใช้ในปาสคาล ตอนที่ 1



เขียนโปรแกรมอ่านภาพ PCX มาแสดงบนจอภาพ และเก็บข้อมูลภาพในโครงสร้างใหม่ แบบไม่มีการบีบอัดข้อมูล

ไฟล์ภาพ PCX เป็นไฟล์ภาพแบบหนึ่งที่เราสามารถนำมาใช้งานบนโปรแกรมที่เขียนด้วยภาษาปาสคาลได้ (ทำงานบนจอสีเท่านั้น) บทความชุดนี้มี 2 ตอนคือ

ตอนที่ 1 กล่าวถึงการเขียนโปรแกรมเพื่ออ่านภาพ PCX มาแสดงบนจอภาพ และทำการเก็บข้อมูลภาพในโครงสร้างใหม่แบบไม่มีการบีบอัดข้อมูล โดยการเก็บภาพใหม่นี้เราสามารถกำหนดขอบเขตของภาพที่จะเก็บได้ พร้อมกับเขียนโปรแกรมอ่านภาพใหม่นี้มาแสดงบนจอภาพแบบง่าย ๆ ซึ่งกำหนดให้ไปแสดงที่ตำแหน่งอื่นของจอภาพได้ (ภาพใหม่นี้ผู้เขียนให้เป็นภาพ *.PIC)

ตอนที่ 2 ผู้เขียนจะแสดงเทคนิคการแสดงภาพ (*.PIC) ให้มีรูปแบบต่าง ๆ กันเกือบ 6 แบบ เช่น ให้กวาดภาพจากมุมบนซ้ายไปจบที่มุมล่างขวา กวาดภาพจากมุมล่างซ้ายไปจบที่มุมบนขวา แสดงแบบสุ่มภาพ ฯลฯ ซึ่งเรามักพบเห็นในโปรแกรมต่างประเทศ

การแสดงภาพแบบที่จะกล่าวถึงในตอนที่ 2 (ฉบับต่อไป) นั้น ผู้เขียนได้นำมาบรรจุไว้ในโปรแกรมไทยทัศน์ 2.0 และโปรแกรมกราฟิก THAISHOW รุ่น 3.0 (ที่พัฒนาล่าสุดเมื่อ 14 เมษายน 2538)

โปรแกรมกราฟิก THAISHOW รุ่น 3.0 ที่ผู้เขียนเพิ่มเติมครั้งล่าสุดนี้ สามารถอ่านไฟล์เนื้อเรื่องได้ยาวขึ้นกว่าเดิม คือสามารถเขียนเนื้อเรื่องได้ยาวถึง 65535x2 =131070 ไบต์ โดยเพิ่มคำสั่ง "ไฟล์ที่สอง (ชื่อไฟล์ที่สอง)" ไว้ที่ท้ายไฟล์แรก ดังนั้นแฟนไทยโชว์ท่านใดที่จำเป็นต้องเขียนเนื้อเรื่องยาว ๆ ให้ติดต่อสอบถามเกี่ยวกับซอฟต์แวร์นี้ที่ผู้เขียนโดยตรง

สำหรับซอฟต์เมล์ของบทความตอนที่ 1 นี้มี 4ไฟล์คือ pcx.pas, sub_pcx.pas, readpic1.pas vgadrv.obj และไฟล์ภาพ view.pcx, train.pcx อีก 2 ไฟล์ ท่านที่ไม่มีเวลานั่งคีย์โปรแกรมก็ติดต่อใช้บริการได้เช่นเคย

เรามาศึกษาการเขียนโปรแกรมกันเลย

ขอบคุณ

ต้นฉบับไฟล์การอ่านไฟล์ภาพ PCX ผู้เขียนได้จากซอฟต์เมล์ของวารสารไมโครคอมพิวเตอร์ ฉบับที่ 99 ของบทความเรื่อง "เทคนิควิธีการนำฟอนต์ CU Writer และภาพ .PCX มาแสดงบนกราฟิกสี" ในหน้าที่ 249-254 เขียนโดย คุณชูเกียรติ แสงมณี และ ส.อ.บุญเรือน คงเขียว (ปัจจุบันทราบว่าเป็น ร.ต.) จึงขออนุญาตและขอบคุณผู้เขียนทั้งสองมาพร้อมนี้ด้วย โดยผู้เขียนขอนำเสนอแนวการโอนภาพในโครงสร้างใหม่เพื่อนำไปใช้งานด้านอื่น ๆ ตามที่กล่าวไว้ข้างต้นต่อไป

ผู้อ่านจะย้อนไปเปิดวารสารฯ ฉบับดังกล่าวเพื่ออ่านรายละเอียดอีกครั้งก็ได้

เพื่อให้การเขียนโปรแกรมเป็นไปตามแนวที่ผู้เขียนถนัด จึงขออนุญาตดัดแปลงโปรแกรมจากของเดิมไปอีกแนวหนึ่ง ซึ่งช่วยให้การแสดงภาพเป็นไปอย่างรวดเร็วยิ่งขึ้น ความเร็วที่เพิ่มขึ้นนี้เกิดจากการเปลี่ยนแปลงการอ่าน

ข้อมูลครั้งละหนึ่งไบต์มาเป็นครั้งละหลายหมื่นไบต์ (ความเร็วที่เพิ่มขึ้นนี้ ผู้อ่านที่ไม่ได้ใช้บริการซอฟต์เมล์ของฉบับที่ 99 คงไม่สามารถเปรียบเทียบได้)

ส่วนสำคัญที่ควรรู้

ไฟล์ภาพ PCX นี้เป็นไฟล์ของบริษัท Zsoft ซึ่งมีใช้ในโปรแกรมต่าง ๆ เช่น ไฟล์ภาพที่ได้จากการสแกนภาพ โปรแกรมพีซีเพนต์บรัช (PC Paintbrush) ฯลฯ โดยไฟล์ภาพชนิด 16 สีจะมีส่วนหัวของไฟล์ 128 ไบต์ ส่วนถัดมาจะเป็นข้อมูลของภาพ

ส่วนหัว 128 ไบต์นี้มีที่จำเป็นต้องนำมาใช้งานดังรูปที่ 1

ส่วนของข้อมูลภาพนั้นเป็นการเก็บแบบเข้ารหัส มีสิ่งควรรู้ดังนี้

1. ข้อมูลที่เรียงลำดับติดต่อกันและเหมือนกันจะถูกนับจำนวนการซ้ำกัน จำนวนการซ้ำนี้ต้องมีค่าไม่มากกว่า 63 จำนวนนับที่มากกว่าก็จะถูกแบ่งเป็นชุด ๆ ละ 63 ไบต์ (หรือเศษของ 63) นำจำนวนนับนี้มาบวกกับ 192 นำมาเป็นข้อมูลไบต์แรกและข้อมูลภาพเป็นข้อมูลที่ 2 ตัวอย่างเช่น

ข้อมูลภาพเป็น 207 อยู่ติดต่อกัน 153 ไบต์ จะถูกแบ่งเป็น 3 ชุดคือ 63, 63, 27 เก็บเป็นข้อมูลภาพได้ดังนี้ 63+192, 207, 63+192, 207, 27+192, 207 ซึ่งจะได้ข้อมูลจริง 255, 207, 255, 207, 219, 207 ทำให้ลดการเก็บภาพได้ 6-153=-147 ไบต์ (ค่าเป็นลบจะลดลง ค่าเป็นบวกจะเพิ่มขึ้น)

ข้อมูลภาพเป็น 23 จำนวน 87 ไบต์ แบ่งเป็น 2 ชุดคือ 63, 24 เก็บข้อมูล 63+192, 23, 24+192, 23 หรือ 255, 23, 216, 23 ขนาดภาพที่เก็บลดลงได้ 4-87=-83 ไบต์

ข้อมูลภาพเป็น 230 (มากกว่า 192) จำนวน 1 ไบต์ ได้ข้อมูล 1 ชุดคือ 1 เก็บข้อมูล 1+192, 230 หรือ 193, 230 ทำให้การเก็บเพิ่มขึ้น 2-1=1 ไบต์

ข้อมูลภาพเป็น 117 (น้อยกว่า 192) จำนวน 2 ไบต์ ได้ข้อมูล 1 ชุดคือ 2 เก็บข้อมูล 2+192, 117 หรือ 194, 117 ทำให้การเก็บเท่าเดิม

ข้อมูลภาพเป็น 117 (น้อยกว่า 192) จำนวน 1 ไบต์ เก็บข้อมูลนี้ลงไปตรง ๆ เลย จึงไม่มีการเปลี่ยนแปลงข้อมูลของการเก็บ คือเก็บ 117 เลย

นี่คือหลักการเก็บภาพจากจอกราฟิกลงในไฟล์ *.PCX โดยแบ่งการเก็บภาพเป็น 4 แพลน

2. การอ่านข้อมูลภาพ (*.PCX) เพื่อคืนกลับสู่จอก็จะใช้แนวคิดย้อนกลับ โดยการตรวจสอบข้อมูลครั้งละ 1-2 ไบต์ (แล้วแต่ว่าไบต์แรกมีค่าเท่าใด)

2.1 ถ้าไบต์แรกมีค่ามากกว่า 192 จะต้องอ่านข้อมูลไบต์ถัดมา (อ่าน 2 ไบต์) นำค่าไบต์แรกมาลบด้วย 192 เป็นจำนวนนับ ไบต์ที่สองจะเป็นข้อมูลภาพ

2.2 ถ้าไบต์แรกมีค่าน้อยกว่าหรือเท่ากับ 192 ไบต์นี้จะเป็นข้อมูลภาพเลย (ทำให้จำนวนนับเป็น 1) กรณีนี้จะอ่านเพียงหนึ่งไบต์

ไบต์ที่	ขนาด (ไบต์)	รายละเอียดค่าที่เก็บ
0	1	รหัสของบริษัท Zsoft
6-7	2	ค่าบนสุดของแกน Y(y1)
10-11	2	ค่าล่างสุดของแกน Y(y2)
16-63	48	ค่าแพเลตต์ของภาพ
65	1	จำนวนแพลนสูงสุด
66-67	2	จำนวนคอลัมน์สูงสุด

รูปที่ 1 ตารางแสดงรายละเอียดส่วนหัวของไฟล์ภาพ PCX

2.3 นำข้อมูลภาพและจำนวนนับที่ได้ไปแสดงบนจอภาพ ตัวอย่างเช่น

ข้อมูลภาพเป็น 117, 255, 207, 219, 207, 231, 92, 45, 194, 39 จะถูกแบ่งการอ่านเป็น 6 ชุดคือ

ชุดที่ 1 117 (จำนวนนับ 1 ข้อมูลภาพ 117)

ชุดที่ 2 255, 207 (จำนวนนับ 255-192=63 ข้อมูลภาพ 207)

ชุดที่ 3 219, 207 (จำนวนนับ 219-192=27 ข้อมูลภาพ 207)

ชุดที่ 4 231, 92 (จำนวนนับ 231-192=39 ข้อมูลภาพ 92)

ชุดที่ 5 45 (จำนวนนับ 1 ข้อมูลภาพ 45)

ชุดที่ 6 194, 39 (จำนวนนับ 219-192=27 ข้อมูลภาพ 39)

โปรแกรมเริ่มต้น

เรามาเริ่มต้นที่การเข้าสู่กราฟิกและการเซตแพเลตต์ของภาพ PCX กันเป็นอันดับแรก โดยการเข้าสู่กราฟิกนี้เราจะเปลี่ยนไฟล์ egavga.bgi ให้เป็นไฟล์ vgadrv.obj ก่อน โดยใช้ไฟล์ binobj.exe ของปาสคาลที่ท่านมีอยู่ดังนี้

```
binobj egavga.bgi vgadrv egavgadriver
```

ทำให้ได้ไฟล์ vgadrv.obj และ egavgadriver เป็นชื่อชุดคำสั่งที่เรียกใช้ผ่านปาสคาล

ไฟล์ sub_pcx.pas เป็นชุดคำสั่งเข้ากราฟิกโหมดและการเซตแพเลตต์ภาพ ซึ่งผู้เขียนจะนำไปใช้ในตอนที่ 2 ด้วย เป็นดังรูปที่ 2 มีรายละเอียดดังนี้

1. บรรทัดที่ 1-2 ชุดคำสั่งเรียกใช้ไฟล์ vgadrv.obj เพื่อเข้ากราฟิก

2. บรรทัดที่ 3-18 ชุดคำสั่งเข้าสู่กราฟิกโหมด โดยจะทำงานเฉพาะจอสีหรือจอวีจีเอโมโนโครมด้วยกราฟิกขนาด 640x480 จุด

3. บรรทัดที่ 19-35 ส่วนการเซตแพเลตต์ของภาพ PCX โดยดำเนินการผ่านอินเตอร์รัพต์หมายเลข $10 (เลขฐานสิบหก) ซึ่งเป็นส่วนควบคุมการแสดงผลบนจอภาพ โดยนำข้อมูลจำนวน 48 ไบต์ (ไบต์ที่ 16-63) ของส่วนหัวภาพมาดำเนินการ โดยการนำข้อมูลมาชุดละ 3 ไบต์ (ค่าของแม่สี 3 สี) มากำหนดผ่านรีจิสเตอร์ที่ใช้งานคือ Regs.DH, Regs.CH และ Regs.CL โดยให้ค่า Regs.AX=$1010 และ Regs.BX เปลี่ยนค่าไปครั้งละ 1

```
1: PROCEDURE EgaVgaDriver; external;
2:   $L Vgadrv.obj

3: PROCEDURE OpenGraph;
4: var
5:   GrDriver,GrMode,Graphck :integer;
6: begin
7:   DetectGraph(GrDriver,GrMode);
8:   if GrDriver=7 then
9:    begin
10:     clrscr; writeln('thai program for VGA only'); halt;
11:   end
12:   else
13:    begin
14:     GrMode:=2;  GrDriver:=9;
15:    end;
16:  Graphck:=RegisterBgidriver(@EgavgaDriver);
17:  InitGraph(GrDriver,GrMode,'');
18: end;

19: PROCEDURE SetPCXPalette;
20: var Count,Xcount:byte;
21: begin
22:  Count:=1; Xcount:=0; Regs.AX:=$1010;
23:  repeat
24:   Regs.DH:=PCXPalette[Count] shr 2;
25:   Regs.CH:=PCXPalette[Count+1] shr 2;
26:   Regs.CL:=PCXPalette[Count+2] shr 2;
27:   Regs.BX:=Xcount; intr($10,Regs);
28:   inc(Count,3); inc(Xcount);
29:  until Count>49;
30:    Regs.AX:=$1000; Regs.BX:=0;
31:   for Count:=0 to 15 do
32:    begin
33:     intr($10,Regs); Regs.BX:=Regs.BX+$0101;
34:    end;
35: end;
```

รูปที่ 2
ชุดคำสั่งเข้ากราฟิกสี
และส่วนการเซตแพเลตต์ภาพ

ไฟล์ sub_pcx.pas นี้จะถูกเรียกใช้ในคราวต่อไปด้วยการกำหนด $I sub_pcx ในไฟล์นั้น ๆ

แสดงภาพหนึ่งแถว

การแสดงภาพโดยใช้ข้อมูลที่ถูกเก็บไว้ในไฟล์ PCX นั้น ถูกกำหนดด้วยหน่วยความจำขนาด 80x4=320 ไบต์ในแต่ละแถวของภาพ โดยแบ่งเป็น 4 แพลนภาพ (แพลนหมายเลข 1, 2, 4 ,8) ซึ่งควบคุมสีของภาพในแต่ละแถวนั้นให้ตรงกับต้นฉบับ

ในหนึ่งจอภาพนั้นประกอบด้วย

1. จำนวนแถวของภาพ 0-479 (ภาพ PCX จะกระทำกับจอภาพขนาด 640x480 จุด)

2. จำนวนไบต์ในหนึ่งแถวมี 4 ชุด ๆ ละ 80 ไบต์ (ลำดับที่ของไบต์แต่ละชุดคือ 0-79) โดยแต่ละชุดนั้นเป็นข้อมูลของแพลนหมายเลข 1, 2, 4 และ 8 ตามลำดับ

3. การแสดงภาพจะเริ่มต้นที่แถวบนสุดของจอภาพก่อน เมื่อให้ข้อมูลแต่ละแถวครบ 4 แพลนจะเลื่อนลงไปยังแถวที่อยู่ถัดลงมา และจบลงเมื่อมาถึงแถวสุดท้ายที่ล่างสุด

ต่อไปนี้เป็นส่วนของไฟล์ pcx.pas โดยเริ่มต้นกันที่โปรแกรมในรูปที่ 3 ก่อน มีรายละเอียดดังนี้

1. บรรทัดที่ 2 เรียกใช้ยูนิตที่เกี่ยวข้อง

2. บรรทัดที่ 3-4 กำหนดชนิดของข้อมูลใหม่ โดยเป็นข้อมูลที่ใช้หน่วยความจำฮีพขนาด 65535 ไบต์

3. บรรทัดที่ 5-11 ตัวแปรที่เกี่ยวข้องมีดังนี้

RowOfPic จำนวนแถวของภาพ

Column, Row, MaxColumn, MaxRow ตัวแปรใช้บอกตำแหน่งของภาพในและคอลัมน์และแถว

PicCol ข้อมูล 4 ชุด ๆ ละ 80 ไบต์ เพื่อรองรับข้อมูลภาพ PCX ที่ต้องการแสดงผล

PCXPalette ข้อมูลเพื่อนำไปเซตแพเลตต์ให้ตรงกับภาพที่อ่านเข้ามา

Pic ตัวแปรฮีพเพื่อรองรับการอ่านข้อมูลจากไฟล์ภาพ (ทำให้การแสดงภาพมีความไวขึ้น)

SizePCX ขนาดความยาวของไฟล์ภาพ PCX

Data ข้อมูลภาพที่จะนำมาแสดงบนจอภาพ

4. บรรทัดที่ 12 เรียกใช้ไฟล์ sub_pcx.pas มาทำงานร่วมด้วย

```
1: PROGRAM PCX_BY_ARTHAN;
2: uses crt,dos,graph;
3: Type PicH=^picHeap;
4:      PicHeap=array[1..65535] of byte;
5: var  RowOfPic:array [0..479] of pointer;
6:      Column,Row,MaxColumn,MaxRow:word;
7:      PicCol:array [1..4,0..79] of byte;
8:      PCXPalette:array[1..48] of byte;
9:      Regs:registers; ch:char;
10:      Pic:PicH; Plane,Data:byte;
11:      SizePCX:longInt;
12: $I Sub_PCX

13: PROCEDURE DisplayPCX(Max:byte);
14: const PlaneNo:array[1..4] of byte=(1,2,4,8);
15: var k,kk:byte;
16: begin
17:  k:=1;
18:  repeat
19:   PicCol[Plane][Column]:=Data; inc(Column);
20:   if Column=MaxColumn then
21:    begin
22:     inc(Plane); Column:=0; kk:=0;
23:     if Plane=5 then
24:      begin
25:       repeat
26:        inc(kk); port[$3C4]:=2; port[$3C5]:=PlaneNo[kk];
27:        move(addr(PicCol[kk])^,RowOfPic[Row]^,MaxColumn);
28:       until kk=4;
29:        Plane:=1; inc(Row);
30:        if Row>MaxRow then
31:          begin
32:            ch:=#27; k:=max; SizePCX:=0;
33:          end;
34:      end;
35:    end;
36:   inc(k);
37:  until k>max;
38: end;
```

รูปที่ 3
ตัวแปรใช้งานและชุดคำสั่งนำข้อมูลภาพไปแสดง

5. บรรทัดที่ 13-38 ชุดคำสั่งแสดงภาพในหนึ่งแถว

6. บรรทัดที่ 14 กำหนดหมายเลขของแพลน เพราะว่าข้อมูลที่เรากำหนดนั้นเป็นข้อมูลชุดที่ 1-4 จึงไม่สามารถนำมากำหนดได้โดยตรง จึงต้องให้ค่าคงที่เป็นหมายเลขแพลนที่ถูกต้องไว้ใช้งาน

7. บรรทัดที่ 19 นำข้อมูลที่อ่านได้ใส่ลงในตัวแปร PicCol ในชุดและคอลัมน์ที่คำนวณได้ พร้อมกับให้เพิ่มค่าตัวแปร Column อีก 1

8. บรรทัดที่ 20-35 ถ้าเลื่อนคอลัมน์มาถึงขวาสุด จะเลื่อนไปยังข้อมูลชุดถัดไป

9. บรรทัดที่ 23-34 เมื่อได้ข้อมูลของแต่ละแถวครบ 4 ชุดแล้ว ก็จะนำข้อมูลทั้งสี่ชุดนี้ไปแสดงบนจอภาพโดยอาศัยพอร์ตหมายเลข $3C4 และ $3C5 โดยใช้คำสั่ง move ข้อมูลทั้งสี่ชุดนี้ (PicCol) ไปบรรจุลงในแถว (RowOfPic) ที่เป็นจริงในขณะนั้นด้วยจำนวนคอลัมน์สูงสุด (MaxColumn) ตามลำดับหมายเลขแพลน

10. บรรทัดที่ 30-33 ส่วนการตรวจสอบการจบการแสดงภาพ

11. การทำงานในส่วนของชุดคำสั่งนี้จะสิ้นสุดเมื่อหมดข้อมูล คือ k>max

อ่านข้อมูลภาพ

ภาพ PCX ที่เรานำมาอ่านในโปรแกรมนี้จะเป็นภาพที่เขียนบนพีซีเพนต์บรัช หรือภาพที่ใช้เครื่องสแกนภาพขึ้นมาก็ได้ ตัวอย่างในซอฟต์เมล์นั้นไฟล์ train.pcx มาจากซอฟต์เมล์ฉบับที่ 99 ส่วนไฟล์ view.pcx นั้นผู้เขียนสแกนขึ้นมาด้วยเครื่องสแกนภาพ Geniscan รุ่น GS-C105 ซึ่งเป็นการสแกนภาพบนดอสธรรมดา (เป็นซอฟต์แวร์ที่ได้มาพร้อมกับการซื้อเครื่องสแกนภาพ)

ชุดคำสั่งในการอ่านภาพจะเป็นดังรูปที่ 4 มีรายละเอียดดังนี้

1. บรรทัดที่ 42-47 ส่วนการแจ้งให้ทราบเมื่อไฟล์ PCX นั้นไม่ใช่ฟอร์แมตภาพที่อ่านได้

2. บรรทัดที่ 48-53 ส่วนการแจ้งเมื่อไม่พบไฟล์ภาพที่กำหนดให้อ่าน

3. บรรทัดที่ 54-60 การอ่านข้อมูลของภาพโดยแยกอ่านมาครั้งละ DataSize ซึ่งในหนึ่งไฟล์อาจมีการอ่านหลายครั้งจนกว่าตัวแปร SizePCX มีค่าเป็นศูนย์

```
39: PROCEDURE ReadPCXFile(FilePCX:string);
40: var Count,Zsoft,NumofPlane:byte;
41:     y1,y2,no,DataSize:word;  PCXFile:file;
42:   Procedure Not_Sixteen_Color;
43:   begin
44:    close(PCXFile); closegraph; sound(330); clrscr; writeln;
45:    writeln('File is not sixteen color or not PCX format file');
46:    delay(1000); nosound; ch:=readkey; halt;
47:   end;
48:   Procedure FileNotFound;
49:   begin
50:    closegraph; sound(330); clrscr; writeln;
51:    writeln('File "',FilePCX,'" not found or disk error');
52:    delay(1000); nosound; ch:=readkey; halt;
53:   end;
54:   Procedure ReadData;
55:    begin
56:      if SizePCX<65534 then DataSize:=SizePCX
57:                       else  DataSize:=65534;
58:      Blockread(PCXFile,Pic^,DataSize);
59:      SizePCX:=SizePCX-DataSize; no:=1;
60:    end;
61: begin
62:  for no:=0 to 479 do RowOfPic[no]:=ptr($A000,no*80);
63:  Row:=0; Column:=0; Plane:=1;
64:  assign(PCXFile,FilePCX);
65:  $I- reset(PCXFile,1); $I+
66:  if IoResult<>0 then FileNotFound;
67:  if FileSize(PCXFile)<128 then Exit;
68:  blockread(PCXFile,Zsoft,1);
69:  Seek(PCXFile,65); blockread(PCXFile,NumofPlane,1);
70:  if (Zsoft<>10) or (NumOfPlane<>4) then Not_Sixteen_Color;
71:  Seek(PCXFile,6);   blockread(PCXFile,Y1,2);
72:  Seek(PCXFile,10);  blockread(PCXFile,Y2,2);
73:  MaxRow:=Y2-Y1;     if MaxRow>479 then MaxRow:=479;
74:  Seek(PCXFile,16);  blockread(PCXFile,PCXPalette,48);
75:  Seek(PCXFile,66);  blockread(PCXFile,MaxColumn,2);
76:  SetPCXPalette;     Seek(PCXFile,128);
77:  SizePCX:=FileSize(PCXFile)-128; Readdata; ch:=#0;
78:  repeat
79:   repeat
80:    data:=Pic^[no]; count:=1;
81:    if Data>192 then
82:     begin
83:      Count:=Data-192; inc(no);
84:      if (no>Datasize) and (SizePCX>0) then ReadData;
85:      data:=Pic^[no];
86:     end;
87:      DisplayPCX(count); inc(no);
88:   until (no>DataSize) or (ch=#27);
89:   if SizePCX>0 then ReadData else ch:=#27;
90:  until ch=#27;
91:  close(PCXFile);
92: end;
```

รูปที่ 4
ชุดคำสั่ง
ในการอ่านไฟล์ภาพ PCX
มาแสดงบนจอกราฟิกสี

4. บรรทัดที่ 62 คำนวณตำแหน่งหน่วยความจำเริ่มต้นของจอภาพในแต่ละแถว

5. บรรทัดที่ 63-67 กำหนดค่าให้ตัวแปร เปิดไฟล์ภาพ ถ้าไม่พบไฟล์ก็แสดงข้อความให้ทราบ หากหาความยาวไฟล์ได้น้อยกว่า 128 ก็จะออกจากชุดคำสั่งนี้ (ป้องกันปัญหาในการอ่านไฟล์ในบรรทัดถัดไป)

6. บรรทัดที่ 68-70 อ่านข้อมูลเพื่อตรวจสอบชนิดของไฟล์ภาพ จำนวนสีของภาพ ซึ่งหากไม่ใช่ภาพที่มี 16 สี จะแจ้งให้ทราบและจบการทำงาน

7. บรรทัดที่ 71-73 อ่านข้อมูลที่เกี่ยวข้องเพื่อคำนวณหาค่าสูงสุดในแกนตั้ง ค่านี้จะต้องไม่มากกว่า 479 ซึ่งเป็นขนาดสูงสุดของกราฟิกที่ใช้

8. บรรทัดที่ 74-75 อ่านค่าแพเลตต์ของภาพและจำนวนคอลัมน์สูงสุดในหนึ่งแถว

9. บรรทัดที่ 76 เซตแพเลตต์ของภาพ และเลื่อนไฟล์ไปยังจุดเริ่มต้นของภาพ

10. บรรทัดที่ 77 หาขนาดของข้อมูลภาพ (ลบขนาดไฟล์ที่อ่านได้ด้วย 128) และอ่านข้อมูลชุดแรก

11. บรรทัดที่ 78-90 วนรอบแสดงภาพจนกว่าจะหมดข้อมูลภาพ

12. บรรทัดที่ 80 ให้ข้อมูลภาพเท่ากับ Pic^[no] โดยให้จำนวนนับเป็นหนึ่ง

13. บรรทัดที่ 81-86 ถ้าข้อมูลภาพมากกว่า 192 (ซึ่งถือว่าเป็น

จำนวนนับ) นำมาลบออกด้วย 192 จะเป็นจำนวนนับ และข้อมูลภาพจะเป็นข้อมูลลำดับถัดไป ทำให้จำนวนนับและข้อมูลภาพเป็นค่าใหม่ ถ้าพบว่าค่าของตัวแปร No>datasize ก็ให้อ่านข้อมูลจากไฟล์ภาพชุดใหม่เข้ามา (No จะเริ่มที่ 1 ใหม่)

14. บรรทัดที่ 87 นำจำนวนนับและข้อมูลไปแสดง (data เป็นตัวแปรโกลบอลจึงไม่ต้องส่งไปด้วย)

15. บรรทัดที่ 88 ถ้ายังมีข้อมูลภาพอีกก็ให้อ่านข้อมูลชุดใหม่เข้ามา (ส่วนนี้เกิดจากคำสั่งบรรทัดที่ 81-86 ไม่จริง ถ้าหมดข้อมูลก็จะจบการทำงานด้วยเงื่อนไข ch=#27

เก็บภาพ *.PIC

การนำภาพ PCX มาเสนอบนจอภาพตามแนวคิดนี้ จะเป็นการนำภาพชนิดที่เก็บเต็มจอกราฟิกมาแสดง ซึ่งในบางครั้งเราไม่มีความต้องการภาพทั้งหมด แต่เราต้องการภาพเพียงบางส่วนเท่านั้น หรือบางทีเราอยากให้ภาพนำมาแสดงทีละส่วนในรูปแบบที่แตกต่างกันออกไป ดังนั้นการใช้ชุดคำสั่งที่กล่าวมาทั้งหมดนี้อาจไม่ตรงกับความต้องการของเรา

ความจริงชุดคำสั่งข้างต้นนี้สามารถดัดแปลงให้แสดงข้อมูลเพียงบางส่วนของภาพได้ โดยการแสดงในแต่ละแถวสามารถกำหนดได้ในขอบเขตภาพที่มีความกว้างครั้งละ 8 จุด เพราะว่าในหนึ่งแถวมีขนาด 80 ไบต์ ดังนั้น 1 ไบต์จึงเป็น 640/80=8 จุด (เริ่มที่ 0, 8, 16, 24, ไปตามลำดับ) ส่วนแถวนั้นเราสามารถกำหนดได้ละเอียดกว่า คือกำหนดค่าใด ๆ ก็ได้ (ในแถวที่ 0-479)

ขอให้ผู้อ่านลองดัดแปลงการอ่านภาพแบบกำหนดบางส่วนดู ซึ่งผู้เขียนขอยืนยันว่าทำได้ โดยเพิ่มตัวแปรอีก 4 ตัว เช่น Column_L, Column_R ค่าคอลัมน์ซ้ายสุดและขวาสุด (ค่า 0-79) และ RowUp, RowDown ค่าแถวบนสุดและล่างสุด(ค่า 0-479) และแทรกคำสั่งแบบตรวจสอบเงื่อนไขอีก 4-5 แห่ง การแสดงภาพแบบเลือกส่วนก็จะทำได้ โดยตำแหน่งภาพยังคงเดิมและอาจแสดงผลช้าลงเนื่องจากช้ากับการตรวจเงื่อนไขที่แทรกเข้ามา

ปัจจุบันเราใช้ฮาร์ดดิสก์กันอยู่เป็นส่วนใหญ่ และราคาถูกมากเมื่อเทียบกับแผ่นดิสก์ เช่น ราคาที่เสนอขายวารสารต่าง ๆ ของขนาด 540 MB (เท่ากับแผ่นดิสก์ 540MB/1.2MB=450 แผ่น=45 กล่อง) ขายในราคาประมาณ 5,200 บาท ขณะที่แผ่นดิสก์ขนาด 1.2 MB ขายกล่องละ 200 บาท ถ้าซื้อให้ได้ขนาดเท่ากับฮาร์ดดิสก์จะต้องใช้เงิน 45x200 =9000 บาท ดังนั้น... ผู้เขียนจึงคิดว่าถ้าเราเก็บภาพจากจอภาพใหม่โดยไม่สนใจกับขนาดของภาพที่อาจโตกว่าต้นฉบับ แต่สนใจความสามารถที่จะนำภาพที่เก็บใหม่นี้มาแสดงในตำแหน่งอื่นบนจอภาพ รวมทั้งสามารถกำหนดรูปแบบการแสดงภาพให้แตกต่างกันออกไปหลาย ๆ แบบ ก็คงมีความคุ้มในการสูญเสียเนื้อที่ในฮาร์ดดิสก์พอสมควร

แนวการเก็บภาพใหม่จะเป็นดังนี้

1. กำหนดขอบเขตของภาพ (มุมบนซ้ายและมุมล่างขวา) เพื่อนำค่าเหล่านี้ไปเก็บไว้ในไฟล์ข้อมูลภาพ

2. เก็บภาพด้วยคำสั่ง getimage โดยคำนึงถึงขนาดของภาพต้องไม่โตกว่า 65,535 ไบต์ด้วย ถ้าโตกว่าก็ต้องแบ่งการเก็บภาพเป็น 2 ส่วน คือส่วนบนและส่วนล่างของภาพ

3. เก็บขนาดของภาพทั้งสองส่วนนั้นลงในไฟล์ข้อมูลภาพ

4. เก็บค่าแพเลตต์ขนาด 48 ไบต์ของภาพ PCX ไว้ในไฟล์ข้อมูลภาพ เพื่อนำมากำหนดโครงสร้างสีให้ตรงกับภาพต้นฉบับ

5. เก็บภาพที่ใช้คำสั่ง getimage ลงในไฟล์ข้อมูลภาพเรียงจากส่วนบนตามด้วยส่วนล่าง

6. ความสูงของภาพที่เก็บกำหนดให้ไม่เกิน 349 แถว เพื่อนำไปใช้ในกราฟิกขนาด 640x350 จุดได้ รวมทั้งจะสามารถนำไปแสดงผลแบบแปลก ๆ บนกราฟิก 640x480 จุดได้ด้วย

กราฟิกขนาด 640x480 จุดนั้น คู่มือการใช้งานเทอร์โบปาสคาลแจ้งว่ามีกราฟิกใช้งานเพียงหน้าเดียว แต่เมื่อใช้งานจริงจะพบว่าในกราฟิกหน้าสองนั้นยังมีที่ว่างอีก 339 แถว (0-338) ซึ่งเราสามารถนำมาใช้งานได้ (มีรายละเอียดในตอนที่ 2)

โปรแกรมในรูปที่ 5 จะเป็นส่วนการเก็บภาพตามแนวคิดใหม่ รายละเอียดของชุดคำสั่งมีดังนี้

1. บรรทัดที่ 96-102 กำหนดให้เก็บภาพในขนาดสูงไม่เกิน 350 แถว เพื่อนำไปใช้ในกราฟิกขนาด 640x350 จุดได้ (ถ้าผู้อ่านต้องการเก็บภาพสูงกว่านี้จะต้องแบ่งภาพเป็น 3 ส่วนและเพิ่มเติมโปรแกรมบางส่วนด้วย)

2. บรรทัดที่ 103-110 คำนวณการแบ่งภาพเป็น 2 ส่วน โดยจะมีส่วนเดียวก็ได้ขึ้นกับค่า SizeUp

3. บรรทัดที่ 111-112 กำหนดการใช้ไฟล์ภาพใหม่ (สกุลภาพเป็น PIC)

4. บรรทัดที่ 116 เก็บข้อมูลของมุมบนซ้าย

5. บรรทัดที่ 117 เก็บข้อมูลของมุมล่างขวา

6. บรรทัดที่ 118 เก็บขนาดของภาพที่สองส่วน

7. บรรทัดที่ 119 เก็บค่าแพเลตต์ของภาพ

8. บรรทัดที่ 120-121 เก็บภาพส่วนบน

9. บรรทัดที่ 122-127 ถ้ามีภาพส่วนล่างให้เก็บภาพส่วนนี้ด้วย

10. บรรทัดที่ 133 เข้าสู่กราฟิก จองหน่วยความจำฮีฟ และอ่านไฟล์ภาพ

11. บรรทัดที่ 134 เก็บภาพตามโครงสร้างใหม่

เมื่อเขียนโปรแกรมสมบูรณ์แล้ว ให้หาไฟล์ภาพ PCX มาทดสอบได้เลย โดยเปลี่ยนชื่อไฟล์ให้ตรงกับที่ท่านมีใช้ แต่ถ้าใช้บริการซอฟต์เมล์ก็ทดสอบอ่านไฟล์ภาพ train.pcx, view.pcx ได้เลย (สองภาพที่ให้มานี้จะมีแพเลตต์ของภาพที่ต่างกัน)

```
93: PROCEDURE SavePic(x1,y1,x2,y2:word; PicName:string);
94: var f:file; SizeUp,SizeDown,Mid,Ydown:word;
95: begin
96:  if (y2-y1)>349 then
97:   begin
98:    setfillstyle(1,1); bar(10,10,400,50);
99:    settextstyle(0,0,2);
100:    outtextxy(25,24,'Error : y2-y1>349');
101:    ch:=readkey; exit;
102:   end;
103:   sizeDown:=0; Ydown:=Y2;
104:   SizeUp:=imagesize(x1,y1,x2,y2);
105:   if SizeUp=0 then
106:    begin
107:     Mid:=(y2-y1) div 2; Ydown:=Y1+Mid;
108:     SizeUp:=imagesize(x1,y1,x2,y1+Mid);
109:     SizeDown:=imagesize(x1,y1+Mid+1,x2,y2);
110:    end;
111:   assign(f,Picname);
112:   $I- rewrite(f); $I+
113:   if IoResult=0 then
114:    begin
115:     reset(f,1);
116:     blockwrite(f,x1,2);      blockwrite(f,y1,2);
117:     blockwrite(f,x2,2);      blockwrite(f,y2,2);
118:     blockwrite(f,SizeUp,2); blockwrite(f,SizeDown,2);
119:     blockwrite(f,PCXPalette,48);
120:     getimage(x1,y1,x2,Ydown,pic^); putimage(x1,y1,pic^,4);
121:     blockwrite(f,pic^,SizeUp);
122:     if SizeDown<>0 then
123:      begin
124:       getimage(x1,y1+Mid+1,x2,Y2,pic^);
125:       blockwrite(f,pic^,SizeDown);
126:       putimage(x1,y1+mid+1,pic^,4);
127:      end;
128:    close(f);
129:   end;
130: end;

131: ==================== PCX.PAS ========================
132: begin
133:  OpenGraph;  new(Pic);  ReadPCXFile('view.pcx');
134:  SavePic(70,3,500,340,'view.pic');
135:  ch:=readkey;  dispose(pic);
136:  closegraph;
137: end.
```

รูปที่ 5
ชุดคำสั่งเก็บภาพ
ในโครงสร้างใหม่
และโปรแกรมหลัก

อ่านภาพ*.PIC

เมื่อมีภาพแล้ว ส่วนต่อไปที่จำเป็นต้องมีก็คือการอ่านภาพนั้นมาแสดงบนจอภาพ โดยเราก็คงเรียงลำดับในการอ่านไฟล์ตามค่าที่เราเก็บไว้ด้วย ซึ่งเราสามารถเปลี่ยนตำแหน่งการแสดงภาพไปยังจุดอื่น ๆ ได้ด้วย

ภาพที่มีแพเลตต์ของภาพที่ต่างกัน ถ้านำมาแสดงบนจอภาพพร้อมกันจะทำให้ภาพอื่นเปลี่ยนแปลงสีไปด้วย ตัวอย่างเช่น เก็บบางส่วนจากไฟล์ภาพ train.pcx, view.pcx ให้เป็น train.pic, view.pic ตามลำดับ ถ้าแสดงภาพเรียงจาก train.pic ไป view.pic ภาพของ train.pic จะมีสีเพี้ยน ทำนองกลับกัน ถ้าแสดง view.pic และ train.pic ก็จะทำให้ภาพ view.pic สีเพี้ยนแทน

โปรแกรมการอ่านภาพ PIC เป็นดังรูปที่ 6 มีรายละเอียดดังนี้

1. บรรทัดที่ 1-7 ส่วนหัวของโปรแกรมซึ่งเรียกใช้ไฟล์ sub_pcx.pas ด้วย

2. บรรทัดที่ 15-16 อ่านค่ามุมทั้งสองของภาพ

3. บรรทัดที่ 18 อ่านค่าแพเลตต์ของภาพและเซตแพเลตต์ด้วยข้อมูลที่อ่านได้

4. บรรทัดที่ 19 กรณีที่กำหนดด้านซ้ายมากเกินไป (ไม่สามารถ putimage ได้) จะคำนวณเลื่อนค่าให้

5. บรรทัดที่ 20 อ่านข้อมูลภาพส่วนบนและแสดงบนจอภาพ

6. บรรทัดที่ 21-26 อ่านข้อมูลภาพส่วนล่าง

7. บรรทัดที่ 33-35 ตัวอย่างการอ่านภาพเดียวกันไปแสดงที่ตำแหน่งต่างกัน

การอ่านภาพแบบนี้ถ้าเป็นการอ่านผ่านฮาร์ดดิสก์จะไม่รู้สึกถึงช่วงเวลาของการแสดงภาพส่วนบนและส่วนล่างมากนัก แต่ถ้าอ่านผ่านแผ่นดิสก์จะสังเกตเห็นการชะงักเนื่องจากคำสั่งในบรรทัดที่ 21-26 ได้ หากต้องการให้แสดงอย่างต่อเนื่องกัน จะต้องสร้างตัวแปรรองรับการอ่านภาพส่วนล่างไว้ด้วย ในฉบับต่อไปจะแก้ไขให้แสดงภาพเป็นอีกแบบหนึ่ง

```
1: PROGRAM READPIC1_BY_ARTHAN;
2: uses crt,dos,graph;
3: Type PicH=^picHeap;
4:      PicHeap=array[1..65535] of byte;
5: var  PCXPalette:array[1..48] of byte;
6:      Regs:registers;  ch:char; Pic:PicH;
7: $I Sub_PCX

8: PROCEDURE ReadPic(x,y:word; PicName:string);
9: var f:file; SizeUp,SizeDown,Mid,x1,y1,x2,y2:word;
10: begin
11:   assign(f,Picname);
12:   $I- reset(f,1); $I+
13:   if IoResult=0 then
14:    begin
15:     blockread(f,x1,2);      blockread(f,y1,2);
16:     blockread(f,x2,2);      blockread(f,y2,2);
17:     blockread(f,SizeUp,2); blockread(f,SizeDown,2);
18:     blockread(f,PCXPalette,48); SetPCXPalette;
19:     if (getmaxx-x)<(x2-x1) then x:=getmaxx-(x2-x1);
20:     blockread(f,pic^,SizeUp); putimage(x,y,pic^,0);
21:     if SizeDown<>0 then
22:      begin
23:       Mid:=(y2-y1) div 2;
24:       blockread(f,pic^,Sizedown);
25:       putimage(x,y+mid+1,pic^,0);
26:      end;
27:    close(f);
28:   end;
29: end;

30: ================== READPIC1.PAS ========================
31: begin
32:  OpenGraph;    new(Pic);
33:  readpic(10,210,'view.pic');
34:  readpic(150,0,'view.pic');
35:  readpic(320,90,'view.pic');
36:  ch:=readkey; dispose(pic);
37:  closegraph;
38: end.
```

รูปที่ 6
โปรแกรมอ่านภาพ
ตามโครงสร้างใหม่

ส่งท้าย

สำหรับการเก็บภาพแบบบีบอัดข้อมูลให้มีขนาดเล็กลงนั้น หากผู้เขียนมีเวลาจะพยายามนำมาเสนอในรับรู้กันต่อไป แต่ในตอนนี้คงต้องรบกวนเนื้อที่ในฮาร์ดดิสก์ไปก่อน

ในตอนที่สองของบทความนี้ ผู้อ่านจะพบกับการนำภาพ PIC มาแสดงในรูปแบบใหม่คือ

1. แสดงแบบกวาดจากมุมบนซ้ายไปจบที่มุมล่างขวา
2. แสดงแบบกวาดจากมุมล่างซ้ายไปจบที่มุมบนขวา
3. แสดงแบบสุ่มภาพส่วนเล็ก ๆ มาแสดงทีละส่วนจนครบทั้งภาพ
4. แสดงแบบเปิดม่าน และแบบปิดม่าน
5. แสดงแบบเลื่อนภาพออกจากตรงกลางให้ใหญ่ขึ้นจนแสดงเต็มภาพ

การเขียนโปรแกรมจะเป็นอย่างไรนั้นคงต้องติดตามในฉบับต่อไป ผู้อ่านท่านใดที่มีความสามารถในการเขียนโปรแกรมก็ทดลองเขียนด้วยตนเองก่อน โดยใช้หลักการดังนี้ "กำหนดกราฟิกหน้าแรก (0) และอ่านภาพไปยังกราฟิกหน้าที่สอง (1) แล้วโอนภาพจากกราฟิกหน้าที่สองมายังหน้าแรก ตามแนวคิดของการโอนภาพแต่ละแบบโดยทำงานจนกว่าจะโอนภาพมาได้หมด"

สำหรับท่านที่ใจร้อนอยากได้ลิสติงไปศึกษาเลยก็ติดต่อโดยตรงที่ผู้เขียนได้ โดยขอค่าบริการ 100 บาท ท่านจะได้ลิสติงทั้งสองตอนไปศึกษาล่วงหน้าก่อนใคร ๆ

ก่อนจากผู้เขียนเรียนรบกวนแฟนประจำบทความว่า หากเป็นไปได้กรุณาส่งไปรษณีย์บัตรแสดงความคิดเห็นสั้น ๆ เกี่ยวกับการเขียนบทความด้านภาษาปาสคาล และเรื่องที่อยากให้เขียนไปให้ผู้เขียนทราบด้วยก็จะเป็นพระคุณยิ่ง เพราะในช่วงนี้ผู้เขียนไม่แน่ใจว่ายังหลงเหลือแฟนที่ใช้งานโปรแกรมบนดอสกันกี่มากน้อย

คงต้องกล่าวว่า พบกันใหม่ในตอนที่สองครับ